

## CONTENTS

<b>SL.NO.</b>	<b>EXPERIMENT NAME</b>	<b>PAGE NO.</b>
1	<b>INTRODUCTION</b>	1
2	<p><b>Program 1</b>            Develop the HTML page named as “Myfirstwebpage.html”. Add the following tags with relevant content.</p> <ul style="list-style-type: none"> <li>1. Set the title of the page as “My First Web Page”</li> <li>2. Within the body use the following tags:               <ul style="list-style-type: none"> <li>a) Moving text = “Basic HTML Tags”</li> <li>b) Different heading tags (h1 to h6)</li> <li>c) Paragraph</li> <li>d) Horizontal line</li> <li>e) Line Break</li> <li>f) Block Quote</li> <li>g) Pre tag</li> </ul> </li> </ul> <p>Different Logical Style (<b>b</b>, <b>u</b>, <b>&lt;sub&gt;</b>, <b>&lt;sup&gt;</b> etc.)</p>	30
3	<p><b>Program 2</b>            Develop the HTML page named as “Table.html” to display your class time table.</p> <ul style="list-style-type: none"> <li>a) Provide the title as Time Table with table header and table footer, row-span and col-span etc.</li> <li>b) Provide various colour options to the cells (Highlight the lab hours and elective hours with different colours.)</li> <li>c) Provide colour options for rows.</li> </ul>	33
4	<p><b>Program 3</b>            Develop an external style sheet named as “style.css” and provide different styles for h2, h3, hr, p, div, span, time, img &amp; a tags. Apply different CSS selectors for tags and demonstrate the significance of each.</p>	37
5	<p><b>Program 4</b>            Develop HTML page named as “registration.html” having variety of HTML input elements with background colors, table for alignment &amp; provide font colors &amp; size using CSS styles.</p>	41
6	<p><b>Program 5</b>            Develop HTML page named as “newpaper.html” having variety of HTML semantic elements with background colors, text-colors &amp; size for figure, table, aside, section, article, header, footer... etc.</p>	45
7	<p><b>Program 6</b>            Apply HTML, CSS and JavaScript to design a simple calculator to perform the following operations: sum, product, difference, remainder, quotient, power, square-root and square.</p>	50

8	<p><b>Program 7</b></p> <p>Develop JavaScript program (with HTML/CSS) for:</p> <ul style="list-style-type: none"> <li>a) Converting JSON text to JavaScript Object</li> <li>b) Convert JSON results into a date</li> <li>c) Converting From JSON To CSV and CSV to JSON</li> <li>d) Create hash from string using crypto.createHash() method</li> </ul>	55
9	<p><b>Program 8</b></p> <p>a. Develop a PHP program (with HTML/CSS) to keep track of the number of visitors visiting the web page and to display this count of visitors, with relevant headings.</p> <p>b. Develop a PHP program (with HTML/CSS) to sort the student records which are stored in the database using selection sort.</p>	60
10	<p><b>Program 9</b></p> <p>Develop jQuery script (with HTML/CSS) for:</p> <ul style="list-style-type: none"> <li>a) Appends the content at the end of the existing paragraph and list.</li> <li>b) Change the state of the element with CSS style using animate() method</li> <li>c) Change the color of any div that is animated.</li> </ul>	67
11	<p><b>Program 10</b></p> <p>Develop a JavaScript program with Ajax (with HTML/CSS) for:</p> <ul style="list-style-type: none"> <li>a) Use ajax() method (without Jquery) to add the text content from the text file by sending ajax request.</li> <li>b) Use ajax() method (with Jquery) to add the text content from the text file by sending ajax request.</li> <li>c) Illustrate the use of getJSON() method in jQuery</li> </ul> <p>Illustrate the use of parseJSON() method to display JSON values.</p>	70
12	<b>VIVA QUESTIONS AND ANSWERS</b>	75

**Program 1**

1. Develop the HTML page named as “Myfirstwebpage.html”. Add the following tags with relevant content
  1. Set the title of the page as “My First Web Page”
  2. Within the body use the following tags:
    - a) Moving text = “Basic HTML Tags”
    - b) Different heading tags (h1 to h6)
    - c) Paragraph
    - d) Horizontal line
    - e) Line Break
    - f) Block Quote
    - g) Pre tag
    - h) Different Logical Style (<b>, <u>, <sub>, <sup> etc.)

**Program**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>My First Web Page</title>
</head>
<body>
    <!-- Moving text -->
    <marquee>Basic HTML Tags</marquee>

    <!-- Different heading tags -->
    <h1>This is a Heading Level 1</h1>
    <h2>This is a Heading Level 2</h2>
    <h3>This is a Heading Level 3</h3>
    <h4>This is a Heading Level 4</h4>
    <h5>This is a Heading Level 5</h5>
    <h6>This is a Heading Level 6</h6>

    <!-- Paragraph -->
    <p>This is a paragraph. It is a block of text that can span multiple lines. It provides detailed information or content to the reader.</p>

    <!-- Horizontal line -->
    <hr>

    <!-- Line Break -->
    <p>Here is a line break<br>And this is the text after the line break.</p>
```

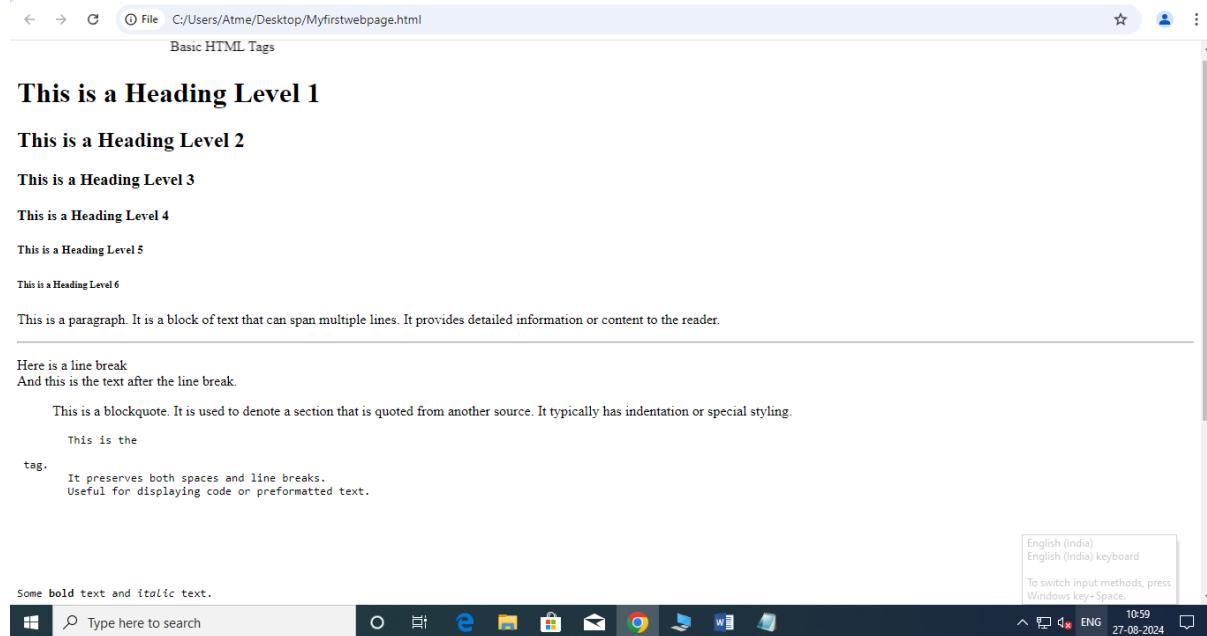
```
<!-- Block Quote -->
<blockquote>
    <p>This is a blockquote. It is used to denote a section that is quoted from another source. It typically has indentation or special styling.</p>
</blockquote>

<!-- Pre tag -->
<pre>
    This is the <pre> tag.
    It preserves both spaces and line breaks.
    Useful for displaying code or preformatted text.
</pre>

<!-- Different Logical Style -->
<p>Some <b>bold</b> text and <i>italic</i> text.</p>
<p>Here is some <u>underlined</u> text.</p>
<p>Here is a subscript: H<sub>2</sub>O</p>
<p>Here is a superscript: E = mc<sup>2</sup></p>

</body>
</html>
```

## Output



**Explanation of the Tags Used:**

1. **<title>**: Sets the title of the page that appears on the browser tab.
2. **<marquee>**: Creates scrolling text (Note: **<marquee>** is obsolete and not recommended for modern web practices, but included here as per the requirement).
3. **<h1> to <h6>**: Represent different levels of headings.
4. **<p>**: Defines a paragraph.
5. **<hr>**: Inserts a horizontal line.
6. **<br>**: Adds a line break.
7. **<blockquote>**: Used for longer quotations or to indicate a block of quoted text.
8. **<pre>**: Displays preformatted text, preserving both spaces and line breaks.
9. **Logical styles**: **<b>**, **<i>**, **<u>**, **<sub>**, and **<sup>** are used for bold, italic, underline, subscript, and superscript text, respectively.

**Program 2**

2. Develop the HTML page named as “Table.html” to display your class time table.
  - a) Provide the title as Time Table with table header and table footer, row-span and col-span etc.
  - b) Provide various colour options to the cells (Highlight the lab hours and elective hours with different colours.)
  - c) Provide colour options for rows.

**Program**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Time Table</title>
    <style>
        table {
            width: 100%;
            border-collapse: collapse;
        }
        th, td {
            border: 1px solid #ddd;
            padding: 8px;
            text-align: center;
        }
        th {
            background-color: #f2f2f2;
        }
        .lab-hour {
            background-color: #f4cccc; /* Light red for lab hours */
        }
        .elective-hour {
            background-color: #d9ead3; /* Light green for elective hours */
        }
        .highlight-row {
            background-color: #f9f9f9; /* Light grey for rows */
        }
        .header {
            background-color: #b6d7a8; /* Light green for header */
            font-weight: bold;
        }
        .footer {
            background-color: #cfe2f3; /* Light blue for footer */
            font-weight: bold;
        }
    </style>
```

```
</head>
<body>
    <h1>Class Time Table</h1>

    <table>
        <!-- Table Header -->
        <thead>
            <tr class="header">
                <th>Time</th>
                <th>Monday</th>
                <th>Tuesday</th>
                <th>Wednesday</th>
                <th>Thursday</th>
                <th>Friday</th>
            </tr>
        </thead>
        <!-- Table Body -->
        <tbody>
            <tr class="highlight-row">
                <td>9:00 - 10:00</td>
                <td class="lab-hour">Math</td>
                <td class="elective-hour">Art</td>
                <td class="lab-hour">Science</td>
                <td class="elective-hour">PE</td>
                <td class="lab-hour">English</td>
            </tr>
            <tr class="highlight-row">
                <td>10:00 - 11:00</td>
                <td class="elective-hour">History</td>
                <td class="lab-hour">Math</td>
                <td class="elective-hour">Computer Science</td>
                <td class="lab-hour">History</td>
                <td class="elective-hour">Art</td>
            </tr>
            <tr class="highlight-row">
                <td>11:00 - 12:00</td>
                <td class="lab-hour">Science</td>
                <td class="elective-hour">English</td>
                <td class="lab-hour">Math</td>
                <td class="elective-hour">Geography</td>
                <td class="lab-hour">Computer Science</td>
            </tr>
            <tr class="highlight-row">
                <td>12:00 - 1:00</td>
                <td colspan="5">Lunch Break</td>
            </tr>
            <tr class="highlight-row">
                <td>1:00 - 2:00</td>
```

```

<td class="elective-hour">Music</td>
<td class="lab-hour">PE</td>
<td class="elective-hour">Math</td>
<td class="lab-hour">Art</td>
<td class="elective-hour">Science</td>
</tr>
<tr class="highlight-row">
<td>2:00 - 3:00</td>
<td class="lab-hour">Computer Science</td>
<td class="elective-hour">History</td>
<td class="lab-hour">English</td>
<td class="elective-hour">Geography</td>
<td class="lab-hour">Math</td>
</tr>
</tbody>
<!-- Table Footer -->
<tfoot>
<tr class="footer">
<td colspan="6">End of Week Timetable</td>
</tr>
</tfoot>
</table>

</body>
</html>

```

## Output

Time	Monday	Tuesday	Wednesday	Thursday	Friday
9:00 - 10:00	Math	Art	Science	PE	English
10:00 - 11:00	History	Math	Computer Science	History	Art
11:00 - 12:00	Science	English	Math	Geography	Computer Science
12:00 - 1:00			Lunch Break		
1:00 - 2:00	Music	PE	Math	Art	Science
2:00 - 3:00	Computer Science	History	English	Geography	Math
End of Week Timetable					



**Explanation of the HTML and CSS:**

1. **<title>**: Sets the title of the page as "Time Table."
2. **Table Header (<thead>)**:
  - o Contains the column titles for the timetable.
3. **Table Body (<tbody>)**:
  - o Contains the timetable data.
  - o Uses `class="highlight-row"` to style alternating rows.
  - o Utilizes classes like `lab-hour` and `elective-hour` to apply different background colors for lab and elective hours.
  - o Uses `colspan` to merge cells across multiple columns for the lunch break.
4. **Table Footer (<tfoot>)**:
  - o Provides a footer row that spans all columns with a summary or note about the table.
5. **CSS Styles**:
  - o Define the appearance of the table, including borders, padding, and background colors for different types of hours and rows.

**Program 3**

3. Develop an external style sheet named as “style.css” and provide different styles for h2, h3, hr, p, div, span, time, img & a tags. Apply different CSS selectors for tags and demonstrate the significance of each.

**Program****style.css**

```
/* Universal selector for basic styling */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

/* Styling for h2 tag */
h2 {
    color: #2c3e50; /* Dark blue color */
    font-family: Arial, sans-serif;
    font-size: 24px;
    margin-bottom: 20px;
}

/* Styling for h3 tag */
h3 {
    color: #16a085; /* Teal color */
    font-family: 'Courier New', Courier, monospace;
    font-size: 20px;
    text-transform: uppercase;
}

/* Styling for horizontal rule */
hr {
    border: none;
    height: 2px;
    background-color: #3498db; /* Blue color */
    margin: 20px 0;
}

/* Styling for p tag */
p {
    color: #34495e; /* Grey color */
    font-family: Georgia, serif;
    font-size: 16px;
    line-height: 1.6;
    margin-bottom: 15px;
```

```
}

/* Styling for div tag */
div {
    background-color: #ecf0f1; /* Light grey background */
    border: 1px solid #bdc3c7; /* Border color */
    padding: 15px;
    margin-bottom: 20px;
    border-radius: 5px;
}

/* Styling for span tag */
span {
    color: #e74c3c; /* Red color */
    font-weight: bold;
}

/* Styling for time tag */
time {
    color: #9b59b6; /* Purple color */
    font-style: italic;
}

/* Styling for img tag */
img {
    max-width: 100%;
    height: auto;
    border: 2px solid #3498db; /* Blue border */
    border-radius: 10px;
}

/* Styling for a tag */
a {
    color: #2980b9; /* Blue color */
    text-decoration: none;
    font-weight: bold;
}

a:hover {
    text-decoration: underline;
    color: #1f618d; /* Darker blue on hover */
}
```

### **Explanation of the CSS Styles and Selectors:**

1. **Universal Selector (\*):**
  - Applies basic margin, padding, and box-sizing to all elements, ensuring consistency across different elements.
2. **Element Selector (`h2, h3, hr, p, div, span, time, img, a`):**
  - Targets specific HTML elements to apply general styling.
3. **`h2`:**
  - Sets color, font-family, font-size, and margin for `h2` headers.
4. **`h3`:**
  - Applies color, font-family, font-size, and text transformation to `h3` headers.
5. **`hr`:**
  - Styles horizontal rules with background color, height, and margin.
6. **`p`:**
  - Defines color, font-family, font-size, line-height, and margin for paragraphs.
7. **`div`:**
  - Adds background color, border, padding, margin, and border-radius to `div` elements.
8. **`span`:**
  - Styles `span` elements with color and font-weight.
9. **`time`:**
  - Applies color and font-style to `time` elements.
10. **`img`:**
  - Sets maximum width, auto height, border, and border-radius for images, ensuring responsiveness.
11. **`a`:**
  - Styles links with color, removes underline by default, and adds bold font weight.
  - `a:hover` pseudo-class changes the link appearance when hovered over, adding underline and changing the color.

### **Applying the Styles**

To use these styles in an HTML document, link the `style.css` file in the `<head>` section of your HTML file:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Styled Page</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h2>Heading Level 2</h2>
  <h3>Heading Level 3</h3>
  <hr>
  <p>This is a paragraph. It has a specific font, color, and line-height.</p>
```

```
<div>
  <p>This is a paragraph inside a div. The div has a background color and border.</p>
  <span>This is a span element with different styling.</span>
</div>
<time datetime="2024-08-27">August 27, 2024</time>

<a href="https://www.example.com">Visit Example.com</a>
</body>
</html>
```

## Output

The screenshot shows a web browser window with the following details:

- Address bar: C:/Users/Atme/Desktop/Web/program3.html
- Content area:
  - Heading Level 2** (Large black font)
  - HEADING LEVEL 3** (Green font)
  - A paragraph: "This is a paragraph. It has a specific font, color, and line-height."
  - A div block with a blue border and white background:
    - "This is a paragraph inside a div. The div has a background color and border."
    - "This is a span element with different styling."
  - An image: A photograph of a path lined with trees.
  - A timestamp: August 27, 2024
  - A link: Visit Example.com

**Program 4**

4. Develop HTML page named as “registration.html” having variety of HTML input elements with background colors, table for alignment & provide font colors & size using CSS styles.

**Program**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Registration Form</title>
    <link rel="stylesheet" href="style.css">
<style>
    body {
        font-family: Arial, sans-serif;
        background-color: #f4f4f4;
        margin: 0;
        padding: 20px;
    }
    table {
        width: 100%;
        max-width: 600px;
        margin: 0 auto;
        border-collapse: collapse;
    }
    th, td {
        padding: 10px;
        text-align: left;
    }
    th {
        background-color: #3498db;
        color: #fff;
        font-size: 18px;
    }
    td {
        background-color: #ecf0f1;
    }
    label {
        display: block;
        margin-bottom: 5px;
        font-size: 16px;
        color: #2c3e50;
    }
    input[type="text"],
    input[type="email"],
```

```
input[type="password"],
select,
textarea {
    width: 100%;
    padding: 8px;
    border: 1px solid #bdc3c7;
    border-radius: 4px;
    font-size: 14px;
}
input[type="submit"] {
    background-color: #2ecc71;
    color: #fff;
    border: none;
    padding: 10px 20px;
    border-radius: 4px;
    font-size: 16px;
    cursor: pointer;
}
input[type="submit"]:hover {
    background-color: #27ae60;
}
.form-container {
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}
.form-title {
    font-size: 24px;
    color: #2c3e50;
    margin-bottom: 20px;
}
</style>
</head>
<body>
<div class="form-container">
    <h1 class="form-title">Registration Form</h1>
    <form action="#" method="post">
        <table>
            <tr>
                <th colspan="2">Personal Information</th>
            </tr>
            <tr>
                <td><label for="first-name">First Name:</label></td>
                <td><input type="text" id="first-name" name="first-name" required></td>
            </tr>
            <tr>
                <td><label for="last-name">Last Name:</label></td>
```

```
<td><input type="text" id="last-name" name="last-name" required></td>
</tr>
<tr>
    <td><label for="email">Email:</label></td>
    <td><input type="email" id="email" name="email" required></td>
</tr>
<tr>
    <td><label for="password">Password:</label></td>
    <td><input type="password" id="password" name="password" required></td>
</tr>
<tr>
    <td><label for="gender">Gender:</label></td>
    <td>
        <select id="gender" name="gender">
            <option value="male">Male</option>
            <option value="female">Female</option>
            <option value="other">Other</option>
        </select>
    </td>
</tr>
<tr>
    <td><label for="bio">Bio:</label></td>
    <td><textarea id="bio" name="bio" rows="4"></textarea></td>
</tr>
<tr>
    <td colspan="2">
        <input type="submit" value="Register">
    </td>
</tr>
</table>
</form>
</div>
</body>
</html>
```

## Output

The screenshot shows a web browser window with the title 'Registration Form'. The form is contained within a div with the class 'form-container'. It features a table layout with rows for personal information. The first row has labels 'First Name:' and 'Last Name:' followed by input fields. The second row has labels 'Email:' and 'Password:' followed by input fields. The third row has a label 'Gender:' followed by a dropdown menu set to 'Male'. The fourth row has a label 'Bio:' followed by a text area. At the bottom is a green 'Register' button. The browser's address bar shows the local file path 'C:/Users/Atme/Desktop/Web/registration.html'. The taskbar at the bottom of the screen displays various application icons.

## Explanation of the HTML and CSS:

### 1. HTML Structure:

- The form is placed inside a `div` with the class `form-container` for styling.
- A table is used for layout alignment, with each row containing labels and corresponding input fields.
- Various input types (`text`, `email`, `password`, `select`, `textarea`) are included to demonstrate different form elements.

### 2. CSS Styles:

- `body`: Sets a background color and padding for the entire page.
- `table`: Centers the table and sets its width. Uses `border-collapse` to ensure borders are collapsed into a single line.
- `th`: Styles table headers with a background color and white text.
- `td`: Styles table data cells with a light grey background.
- `label`: Styles labels with font size and color.
- `input`, `select`, `textarea`: Applies styles to form elements including padding, border, and font size.
- `input[type="submit"]`: Styles the submit button with a green background, white text, and hover effect.
- `.form-container`: Adds padding, background color, border radius, and box shadow to the form container for a card-like appearance.
- `.form-title`: Styles the form title with font size and color.

**Program 5**

5. Develop HTML page named as “newspaper.html” having variety of HTML semantic elements with background colors, text-colors & size for figure, table, aside, section, article, header, footer... etc.

**Program**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Newspaper Layout</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 20px;
        }
        header {
            background-color: #2c3e50;
            color: #ecf0f1;
            padding: 20px;
            text-align: center;
        }
        footer {
            background-color: #2c3e50;
            color: #ecf0f1;
            padding: 10px;
            text-align: center;
            position: fixed;
            bottom: 0;
            width: 100%;
        }
        main {
            display: flex;
            gap: 20px;
            margin-bottom: 60px; /* To ensure footer does not overlap content */
        }
        article {
            flex: 2;
            background-color: #fff;
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
        }
    </style>

```

```
aside {  
    flex: 1;  
    background-color: #ecf0f1;  
    padding: 20px;  
    border-radius: 8px;  
}  
section {  
    margin-bottom: 20px;  
}  
h1, h2, h3 {  
    color: #34495e;  
}  
table {  
    width: 100%;  
    border-collapse: collapse;  
    margin-top: 20px;  
}  
table, th, td {  
    border: 1px solid #bdc3c7;  
}  
th {  
    background-color: #3498db;  
    color: #fff;  
    padding: 10px;  
}  
td {  
    background-color: #fff;  
    padding: 10px;  
    text-align: left;  
}  
figure {  
    margin: 0;  
    padding: 0;  
}  
figcaption {  
    font-size: 14px;  
    color: #7f8c8d;  
    text-align: center;  
}  
.figure-container {  
    background-color: #fff;  
    padding: 10px;  
    border-radius: 8px;  
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);  
    margin: 20px 0;  
}  
</style>  
</head>
```

```
<body>
  <header>
    <h1>Newspaper Header</h1>
    <p>Your source for the latest news</p>
  </header>

  <main>
    <article>
      <section>
        <h2>Main Article</h2>
        <p>This is the main article section. It contains the primary content of the newspaper page. Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
        <figure class="figure-container">
          
          <figcaption>Figure 1: Example image caption.</figcaption>
        </figure>
        <table>
          <caption>Sample Table</caption>
          <thead>
            <tr>
              <th>Header 1</th>
              <th>Header 2</th>
              <th>Header 3</th>
            </tr>
          </thead>
          <tbody>
            <tr>
              <td>Data 1</td>
              <td>Data 2</td>
              <td>Data 3</td>
            </tr>
            <tr>
              <td>Data 4</td>
              <td>Data 5</td>
              <td>Data 6</td>
            </tr>
          </tbody>
        </table>
      </section>
    </article>

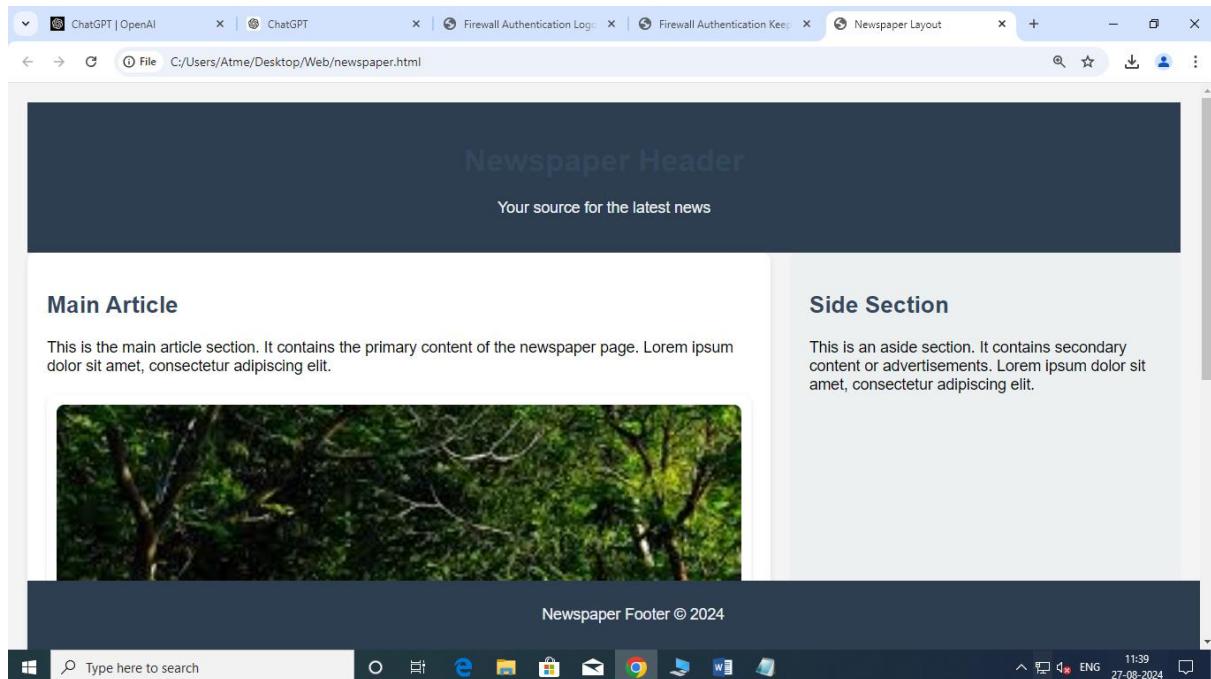
    <aside>
      <h2>Side Section</h2>
      <p>This is an aside section. It contains secondary content or advertisements. Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    </aside>
  </main>
```

```

<footer>
  <p>Newspaper Footer © 2024</p>
</footer>
</body>
</html>

```

## Output



## Explanation of the HTML and CSS:

### 1. HTML Structure:

- **<header>**: Contains the main heading and subtitle of the newspaper.
- **<footer>**: Provides footer information, fixed at the bottom of the page.
- **<main>**: Uses flexbox to layout the `article` and `aside` side by side.
- **<article>**: Represents the main content area of the page.
- **<aside>**: Contains secondary content or sidebar information.
- **<section>**: Used within the `article` to group related content.
- **<figure>** and **<figcaption>**: Used to include an image with a caption.
- **<table>**: Contains tabular data with headers and a caption.

### 2. CSS Styles:

- **body**: Sets a background color, font, and padding for the entire page.
- **header** and **footer**: Define background colors, text colors, and padding. The footer is positioned fixed at the bottom of the page.
- **main**: Utilizes flexbox to create a responsive layout for the main content and sidebar.

- 
- **article** and **aside**: Style the primary content and sidebar with background colors, padding, and shadows.
  - **table**: Styles the table with borders, background colors for headers, and padding.
  - **figure** and **figcaption**: Apply styling to the figure and caption, including background color and text color.

**Program 6**

6. Apply HTML, CSS and JavaScript to design a simple calculator to perform the following operations: sum, product, difference, remainder, quotient, power, square-root and square.

**Program**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Simple Calculator</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            background-color: #f4f4f4;
            margin: 0;
        }
        .calculator {
            background-color: #fff;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
        }
        .calculator h1 {
            font-size: 24px;
            color: #2c3e50;
            margin-bottom: 20px;
        }
        .calculator input,
        .calculator select,
        .calculator button {
            margin-bottom: 10px;
            padding: 10px;
            border: 1px solid #bdc3c7;
            border-radius: 5px;
            font-size: 16px;
            width: calc(100% - 22px); /* Adjust for padding and border */
        }
        .calculator button {
            background-color: #3498db;
            color: #fff;
            border: none;
        }
    </style>
</head>
<body>
    <div class="calculator">
        <h1>Simple Calculator</h1>
        <input type="text" id="display" value="0" style="width: 100%; height: 40px; border: none; border-bottom: 2px solid #ccc; font-size: 18px; font-weight: bold; padding: 0 10px; margin-bottom: 10px;">
        <div>
            <button>AC</button>
            <button>÷</button>
            <button>×</button>
            <button>-</button>
            <button>+</button>
            <button>0</button>
            <button>1</button>
            <button>2</button>
            <button>3</button>
            <button>4</button>
            <button>5</button>
            <button>6</button>
            <button>7</button>
            <button>8</button>
            <button>9</button>
            <button>.(</button>
            <button>=</button>
        </div>
    </div>
</body>
</html>
```

```
        cursor: pointer;
    }
.calculator button:hover {
    background-color: #2980b9;
}
.calculator .result {
    font-size: 18px;
    color: #2c3e50;
    margin-top: 10px;
}
</style>
</head>
<body>
<div class="calculator">
    <h1>Simple Calculator</h1>
    <input type="number" id="num1" placeholder="Enter first number">
    <input type="number" id="num2" placeholder="Enter second number">
    <select id="operation">
        <option value="sum">Sum</option>
        <option value="difference">Difference</option>
        <option value="product">Product</option>
        <option value="quotient">Quotient</option>
        <option value="remainder">Remainder</option>
        <option value="power">Power</option>
        <option value="sqrt">Square Root</option>
        <option value="square">Square</option>
    </select>
    <button onclick="calculate()">Calculate</button>
    <div class="result" id="result"></div>
</div>

<script>
    function calculate() {
        // Get values from inputs
        const num1 = parseFloat(document.getElementById('num1').value);
        const num2 = parseFloat(document.getElementById('num2').value);
        const operation = document.getElementById('operation').value;

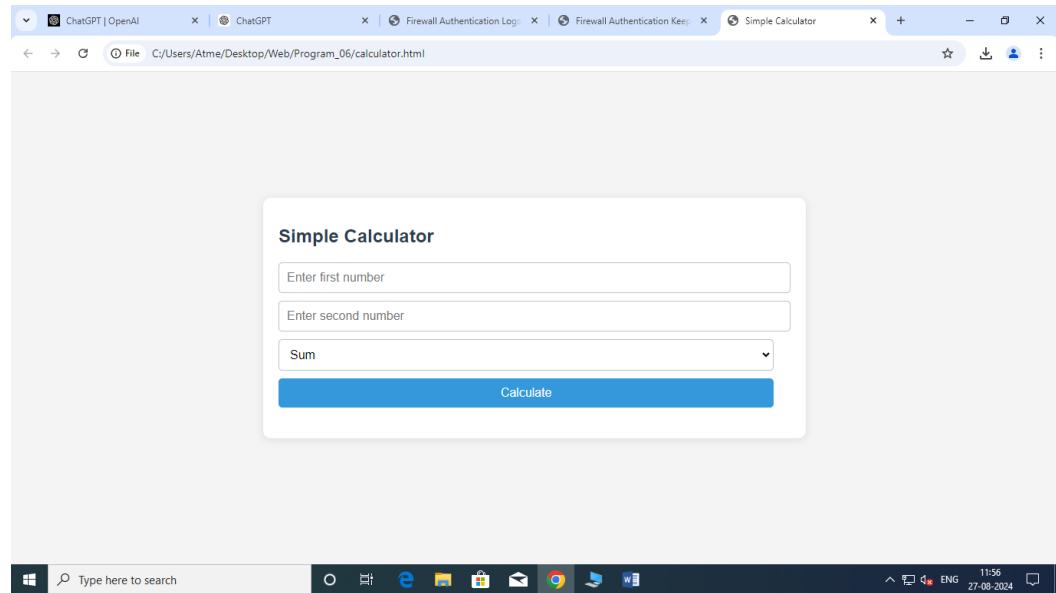
        // Result variable
        let result;

        // Perform calculation based on selected operation
        switch (operation) {
            case 'sum':
                result = num1 + num2;
                break;
            case 'difference':
                result = num1 - num2;
```

```
        break;
    case 'product':
        result = num1 * num2;
        break;
    case 'quotient':
        result = num1 / num2;
        break;
    case 'remainder':
        result = num1 % num2;
        break;
    case 'power':
        result = Math.pow(num1, num2);
        break;
    case 'sqrt':
        result = Math.sqrt(num1);
        break;
    case 'square':
        result = Math.pow(num1, 2);
        break;
    default:
        result = 'Invalid operation';
    }

    // Display result
    document.getElementById('result').textContent = 'Result: ' + result;
}
</script>
</body>
</html>
```

## Output



The image contains six separate screenshots of the "Simple Calculator" application, each showing a different operation:

- Sum:** Shows inputs 10 and 5, operation Sum, result 15.
- Difference:** Shows inputs 10 and 5, operation Difference, result 5.
- Product:** Shows inputs 10 and 5, operation Product, result 50.
- Quotient:** Shows inputs 10 and 5, operation Quotient, result 2.
- Remainder:** Shows inputs 10 and 5, operation Remainder, result 0.
- Power:** Shows inputs 10 and 5, operation Power, result 100000.

Each screenshot shows the same basic interface with input fields for two numbers, a dropdown for the operation, a "Calculate" button, and a "Result" label below it.

**Explanation:****1. HTML:**

- o The HTML structure includes:
  - Two input fields for numbers.
  - A <select> dropdown to choose the operation.
  - A <button> to trigger the calculation.
  - A <div> to display the result.

**2. CSS:**

- o Styles are applied to center the calculator on the page, style the form elements, and add hover effects to the button.
- o The .calculator class styles the container, adding padding, border-radius, and box-shadow for a clean look.

**3. JavaScript:**

- o The calculate function retrieves the values from the input fields and the selected operation from the dropdown.
- o It performs the calculation based on the chosen operation using a switch statement.
- o The result is then displayed in the <div> with the id="result".

This simple calculator performs all the specified operations, and the JavaScript handles the logic for each operation based on user input.

**Program 7**

7. Develop JavaScript program (with HTML/CSS) for:
  - a) Converting JSON text to JavaScript Object
  - b) Convert JSON results into a date
  - c) Converting from JSON to CSV and CSV to JSON
  - d) Create hash from string using crypto.createHash() method

**Program**

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>JSON and CSV Converter</title>
        <style>
            body {
                font-family: Arial, sans-serif;
                margin: 20px;
            }
            .container {
                margin-bottom: 20px;
            }
            textarea, input[type="text"], button {
                display: block;
                width: 100%;
                margin-bottom: 10px;
                padding: 8px;
                box-sizing: border-box;
            }
            button {
                background-color: #4CAF50;
                color: white;
                border: none;
                cursor: pointer;
            }
            button:hover {
                background-color: #45a049;
            }
            .result {
                white-space: pre-wrap;
                background: #f4f4f4;
                padding: 10px;
            }
        </style>
    </head>
    <body>
```

```

<div class="container">
  <h2>Convert JSON to JavaScript Object</h2>
  <textarea id="jsonText" placeholder='Enter JSON here'></textarea>
  <button onclick="convertJsonToObject()">Convert JSON</button>
  <div id="jsonObject" class="result"></div>
</div>

<div class="container">
  <h2>Convert JSON to Date</h2>
  <input type="text" id="jsonDate" placeholder='Enter JSON Date string'>
  <button onclick="convertJsonToDate()">Convert to Date</button>
  <div id="dateResult" class="result"></div>
</div>

<div class="container">
  <h2>Convert JSON to CSV and CSV to JSON</h2>
  <textarea id="jsonCsvText" placeholder='Enter JSON for CSV conversion'></textarea>
  <button onclick="convertJsonToCsv()">JSON to CSV</button>
  <div id="csvResult" class="result"></div>

  <textarea id="csvJsonText" placeholder='Enter CSV text'></textarea>
  <button onclick="convertCsvToJson()">CSV to JSON</button>
  <div id="jsonResult" class="result"></div>
</div>

<div class="container">
  <h2>Create Hash from String</h2>
  <input type="text" id="stringInput" placeholder='Enter string to hash'>
  <button onclick="createHash()">Create Hash</button>
  <div id="hashResult" class="result"></div>
</div>

<script src="app.js"></script>
</body>
</html>

```

### JavaScript (app.js)

Now let's add the JavaScript code to handle each of the tasks.

```

} // Convert JSON text to JavaScript Object
function convertJsonToObject() {
  const jsonText = document.getElementById('jsonText').value;
  try {
    const obj = JSON.parse(jsonText);
    document.getElementById('jsonObject').textContent = JSON.stringify(obj, null, 2);
  } catch (e) {

```

```

        document.getElementById('jsonObject').textContent = 'Invalid JSON';
    }
}

// Convert JSON date string to Date
function convertJsonToDate() {
    const jsonDate = document.getElementById('jsonDate').value;
    try {
        const date = new Date(jsonDate);
        document.getElementById('dateResult').textContent = date.toString();
    } catch (e) {
        document.getElementById('dateResult').textContent = 'Invalid Date';
    }
}

// Convert JSON to CSV
function convertJsonToCsv() {
    const jsonText = document.getElementById('jsonCsvText').value;
    try {
        const data = JSON.parse(jsonText);
        const csv = Object.keys(data[0]).join(',') + '\n' +
            data.map(row => Object.values(row).join(',')).join('\n');
        document.getElementById('csvResult').textContent = csv;
    } catch (e) {
        document.getElementById('csvResult').textContent = 'Invalid JSON';
    }
}

// Convert CSV to JSON
function convertCsvToJson() {
    const csvText = document.getElementById('csvJsonText').value;
    const rows = csvText.split('\n').map(row => row.split(','));
    const headers = rows.shift();
    const json = rows.map(row => {
        let obj = {};
        row.forEach((value, index) => obj[headers[index]] = value);
        return obj;
    });
    document.getElementById('jsonResult').textContent = JSON.stringify(json, null, 2);
}

// Create hash from string
async function createHash() {
    const crypto = window.crypto || window.msCrypto; // For IE11
    const str = document.getElementById('stringInput').value;
    const encoder = new TextEncoder();
    const data = encoder.encode(str);
    const hashBuffer = await crypto.subtle.digest('SHA-256', data);
}

```

```
const hashArray = Array.from(new Uint8Array(hashBuffer));
const hashHex = hashArray.map(b => b.toString(16).padStart(2, '0')).join("");
document.getElementById('hashResult').textContent = hashHex;
```

## Output

**Convert JSON to JavaScript Object**

```
{ "name": "John", "age": 30, "city": "New York"}
```

Convert JSON

```
{
  "name": "John",
  "age": 30,
  "city": "New York"
}
```

**Convert JSON to Date**

08/29/2024

Convert to Date

Thu Aug 29 2024 00:00:00 GMT+0530 (India Standard Time)

**Convert JSON to CSV and CSV to JSON**

```
[{"name": "john", "age": 30}, {"name": "Jane", "age": 25}]
```

JSON to CSV

```
name,age
john,30
Jane,25
```

name,age
John,30
Jane,25

CSV to JSON

Wednesday, August 28, 2024 2:49 PM 8/28/2024

Thu Aug 29 2024 00:00:00 GMT+0530 (India Standard Time)

**Convert JSON to CSV and CSV to JSON**

```
[{"name": "john", "age": 30}, {"name": "Jane", "age": 25}]
```

JSON to CSV

```
name,age
john,30
Jane,25
```

name,age
John,30
Jane,25

CSV to JSON

```
[
  {
    "name": "John",
    "age": "30"
  },
  {
    "name": "Jane",
    "age": "25"
  }
]
```

**Create Hash from String**

ATME

Create Hash

7d9ead2729461686b0844c88bec8ceae1ad7fbeaf23f3fe5c09b590440215436

2:49 PM 8/28/2024

**Explanation:**

1. **Convert JSON to JavaScript Object:** The function `convertJsonToObject` parses the JSON text and displays it in a formatted way. If the JSON is invalid, an error message is shown.
2. **Convert JSON Date to JavaScript Date:** The function `convertJsonToDate` creates a JavaScript `Date` object from the JSON date string and displays it.
3. **Convert JSON to CSV:** The function `convertJsonToCsv` parses JSON data into a CSV format. It assumes that the input JSON is an array of objects.
4. **Convert CSV to JSON:** The function `convertCsvToJson` parses CSV data into JSON. It assumes the first row of the CSV contains headers.
5. **Create Hash from String:** The function `createHash` uses the Web Crypto API to generate a SHA-256 hash of the input string.

**Program 8**

8. a. Develop a PHP program (with HTML/CSS) to keep track of the number of visitors visiting the web page and to display this count of visitors, with relevant headings.  
b. Develop a PHP program (with HTML/CSS) to sort the student records which are stored in the database using selection sort.

**Program 8a**

Develop a PHP program (with HTML/CSS) to keep track of the number of visitors visiting the web page and to display this count of visitors, with relevant headings.

**Step 1: Create the PHP Script**

First, create a PHP file named `program_8a.php`. This script will handle the counting of visitors and display the count on the web page.

```
<?php
// Define the path to the file where the count will be stored
$countFile = 'count.txt';

// Check if the count file exists
if (!file_exists($countFile)) {
    // If the file doesn't exist, create it and initialize the count to 0
    file_put_contents($countFile, '0');
}

// Read the current count from the file
$count = (int)file_get_contents($countFile);

// Increment the count
$count++;

// Save the updated count back to the file
file_put_contents($countFile, $count);

// HTML output
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Visitor Counter</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            display: flex;
            justify-content: center;
            align-items: center;
        }
    </style>

```

```

height: 100vh;
background-color: #f4f4f4;
margin: 0;
}
.container {
    text-align: center;
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}
h1 {
    color: #333;
}
.count {
    font-size: 2em;
    color: #007bff;
}

```

</style>

</head>

<body>

<div class="container">

<h1>Visitor Counter</h1>

<p class="count">You are visitor number: <?php echo \$count; ?></p>

</div>

</body>

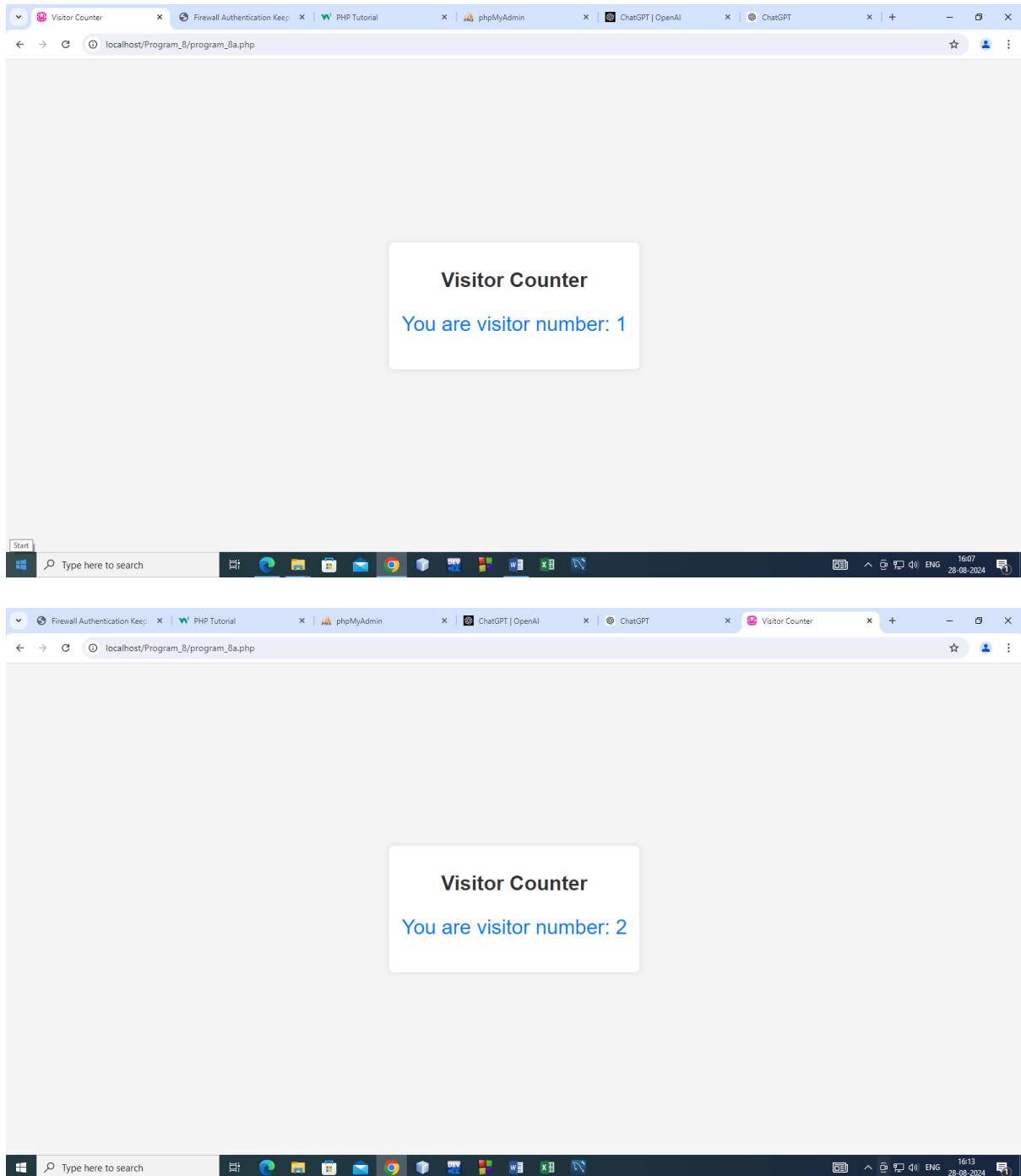
</html>

### Step 2: Explanation

- File Handling:** The PHP script uses `count.txt` to keep track of the number of visitors. If this file does not exist, it creates it and initializes the count to 0. Then, it reads the current count, increments it, and writes the updated count back to the file.
- HTML/CSS:**
  - HTML:** The page has a simple structure with a heading and a paragraph that displays the visitor count.
  - CSS:** Basic styles are applied to center the content and make the page look clean. The container has a box shadow for a subtle effect, and the visitor count is styled to stand out.

### Step 3: Upload and Test

- Upload Files:** Place `program_8a.php` on your web server. Ensure that `count.txt` is writable by the web server (check permissions if needed).
- Access Your Page:** Open your web browser and navigate to the location where `program_8a.php` is hosted. You should see the visitor counter and the count will increment with each page load.

**Output**

## Program 8b

Develop a PHP program (with HTML/CSS) to sort the student records which are stored in the database using selection sort.

### Steps

- Set up your database:** You'll need a MySQL database with a table for student records. For simplicity, let's assume you have a database named `web_lab_students` and a table named `students` with fields `Stu_name`, `USN` and `Sem`.
- Create the PHP script:** This script will connect to the database, fetch student records, sort them using selection sort, and then display the results.
- Write the HTML/CSS:** This part will handle the presentation of the sorted data.

### 1. Database Setup

Assuming your database `web_lab_students` and table `students` are already created, here is an example of how the table might look:

```
CREATE DATABASE web_lab_students;
```

```
USE web_lab_students;
```

```
CREATE TABLE students (
    Stu_Name VARCHAR(100),
    USN VARCHAR(20),
    Sem INT );
```

```
INSERT INTO students (Stu_Name, USN, Sem) VALUES('Ramesh', '4AD10CS001',3)
INSERT INTO students (Stu_Name, USN, Sem) VALUES('Suresh', '4AD10CS005',2)
INSERT INTO students (Stu_Name, USN, Sem) VALUES('Mahesh', '4AD10CS004',8)
INSERT INTO students (Stu_Name, USN, Sem) VALUES('Raveesh', '4AD10CS020',6)
```

Stu_Name	USN	Sem
Ramesh	4AD10CS001	3
Suresh	4AD10CS005	2
Mahesh	4AD10CS004	8
Raveesh	4AD10CS020	6

## 2. PHP Script (with Selection Sort)

Create a file named `sort_students.php` with the following content:

```

<?php
// Database configuration
$host = 'localhost';
$dbname = 'web_lab_students';
$username = 'root'; // Replace with your MySQL username
$password = 'root'; // Replace with your MySQL password

// Create a new PDO instance
try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch (PDOException $e) {
    die("Could not connect to the database: " . $e->getMessage());
}

// Fetch student records from the database
$sql = 'SELECT * FROM students';
$stmt = $pdo->query($sql);
$students = $stmt->fetchAll(PDO::FETCH_ASSOC);

// Function to perform selection sort
function selectionSort(&$array) {
    $n = count($array);
    for ($i = 0; $i < $n - 1; $i++) {
        $minIndex = $i;
        for ($j = $i + 1; $j < $n; $j++) {
            if ($array[$j]['Sem'] < $array[$minIndex]['Sem']) {
                $minIndex = $j;
            }
        }
        if ($minIndex != $i) {
            $temp = $array[$i];
            $array[$i] = $array[$minIndex];
            $array[$minIndex] = $temp;
        }
    }
}

// Sort students by semester using selection sort
selectionSort($students);
?>
<!DOCTYPE html>
<html lang="en">
<head>
```

```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Sorted Student Records</title>
<style>
body {
    font-family: Arial, sans-serif;
    margin: 20px;
    background-color: #f4f4f4;
}
table {
    width: 100%;
    border-collapse: collapse;
}
th, td {
    padding: 10px;
    border: 1px solid #ddd;
    text-align: left;
}
th {
    background-color: #4CAF50;
    color: white;
}
tr:nth-child(even) {
    background-color: #f2f2f2;
}
h1 {
    color: #333;
}
</style>
</head>
<body>
    <h1>Sorted Student Records by Semester</h1>
    <table>
        <thead>
            <tr>
                <th>Name</th>
                <th>USN</th>
                <th>Semester</th>
            </tr>
        </thead>
        <tbody>
            <?php foreach ($students as $student): ?>
            <tr>
                <td><?php echo htmlspecialchars($student['Stu_Name']); ?></td>
                <td><?php echo htmlspecialchars($student['USN']); ?></td>
                <td><?php echo htmlspecialchars($student['Sem']); ?></td>
            </tr>
        <?php endforeach; ?>
```

```
</tbody>
</table>
</body>
</html>
```

## Output

Name	USN	Semester
Suresh	4AD10CS005	2
Ramesh	4AD10CS001	3
Raveesh	4AD10CS020	6
Mahesh	4AD10CS004	8

## Explanation

- Database Connection:** Establish a connection to the MySQL database using PDO.
- Fetching Data:** Retrieve all student records from the `students` table.
- Selection Sort Function:** Implement the selection sort algorithm to sort students by their semester (`sem`).
- HTML and CSS:** Display the sorted student records in an HTML table, styled with basic CSS.

## 3. Running the Script

- Make sure your MySQL server is running, and PHP is properly configured.
- Place the `sort_students.php` file in your web server's root directory.
- Access the script via your browser, for example,  
`http://localhost/sort_students.php`.

This script will display student records sorted by the `sem` field using the selection sort algorithm. Adjust database credentials and table schema as needed.

**Program 9**

9. Develop jQuery script (with HTML/CSS) for:
  - a. Appends the content at the end of the existing paragraph and list.
  - b. Change the state of the element with CSS style using animate() method
  - c. Change the color of any div that is animated.

**Program**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>jQuery Animations and Manipulations</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
            background-color: #f4f4f4;
        }
        .container {
            background-color: #fff;
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
        }
        p, ul {
            margin: 10px 0;
        }
        .animated-div {
            width: 100px;
            height: 100px;
            background-color: #3498db;
            margin-top: 20px;
            transition: background-color 0.5s;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>jQuery Example</h1>

        <!-- Paragraph and List to append content to -->
        <p id="paragraph">This is a paragraph.</p>
        <ul id="list">
```

```
<li>Item 1</li>
<li>Item 2</li>
</ul>

<!-- Buttons for jQuery actions -->
<button id="appendText">Append Text to Paragraph</button>
<button id="appendListItem">Append List Item</button>
<button id="animateDiv">Animate Div</button>

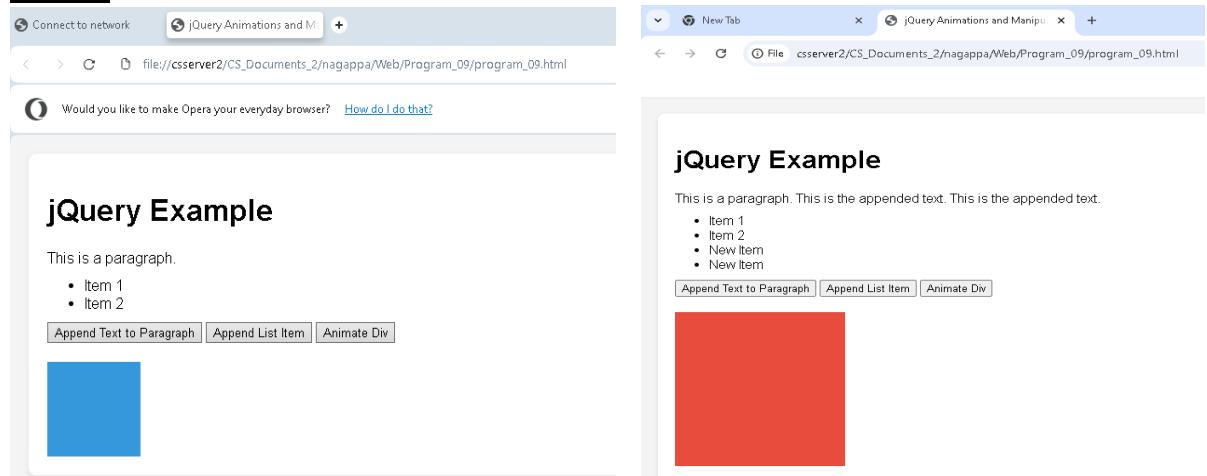
<!-- Div to animate -->
<div id="animatedDiv" class="animated-div"></div>
</div>

<!-- jQuery Library -->
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
$(document).ready(function() {
    // Append content to paragraph and list
    $('#appendText').click(function() {
        $('#paragraph').append(' This is the appended text.');
    });

    $('#appendListItem').click(function() {
        $('#list').append('<li>New Item</li>');
    });

    // Animate div
    $('#animateDiv').click(function() {
        $('#animatedDiv').animate({
            width: '200px',
            height: '200px'
        }, 1000, function() {
            // Change color after animation completes
            $(this).css('background-color', '#e74c3c');
        });
    });
});
</script>
</body>
</html>
```

## **Output**



## **Explanation:**

### **1. HTML:**

- Contains a paragraph and a list to which content can be appended.
- Includes buttons for appending text, appending list items, and animating a `div`.
- The `div` with the `id="animatedDiv"` is styled and animated.

### **2. CSS:**

- Basic styling for the body, container, paragraph, and list.
- The `.animated-div` class provides initial styling and a transition effect for background color changes.

### **3. jQuery:**

#### **○ Append Content:**

- `$('#appendText').click():` Appends text to the existing paragraph when the button is clicked.
- `$('#appendListItem').click():` Appends a new list item to the existing list when the button is clicked.

#### **○ Animate Div:**

- `$('#animateDiv').click():` Animates the size of the `div` when the button is clicked. After the animation completes, the background color of the `div` changes to red.

This script demonstrates basic jQuery functionalities for manipulating and animating HTML elements. You can save the above code into an `index.html` file and open it in a web browser to see it in action.

**Program 10**

10. Develop a JavaScript program with Ajax (with HTML/CSS) for:
- Use ajax() method (without Jquery) to add the text content from the text file by sending ajax request.
  - Use ajax() method (with Jquery) to add the text content from the text file by sending ajax request.
  - Illustrate the use of getJSON() method in jQuery
  - Illustrate the use of parseJSON() method to display JSON values.

**Program**

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>AJAX with and without jQuery</title>
        <style>
            body {
                font-family: Arial, sans-serif;
                margin: 20px;
                background-color: #f4f4f4;
            }
            .container {
                background-color: #fff;
                padding: 20px;
                border-radius: 8px;
                box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
            }
            button {
                margin: 10px;
                padding: 10px;
                border: 1px solid #bdc3c7;
                border-radius: 5px;
                background-color: #3498db;
                color: #fff;
                cursor: pointer;
            }
            button:hover {
                background-color: #2980b9;
            }
            pre {
                background-color: #ecf0f1;
                padding: 10px;
                border-radius: 5px;
                white-space: pre-wrap;
```

```
        }
    </style>
</head>
<body>
<div class="container">
<h1>AJAX Examples</h1>

<!-- Buttons for AJAX requests -->
<button id="loadTextWithoutJQuery">Load Text (No jQuery)</button>
<button id="loadTextWithJQuery">Load Text (With jQuery)</button>
<button id="loadJson">Load JSON</button>
<button id="parseJson">Parse JSON</button>

<!-- Output areas -->
<h2>Text File Content</h2>
<pre id="textOutput"></pre>

<h2>JSON Data</h2>
<pre id="jsonOutput"></pre>
</div>

<!-- jQuery Library -->
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<!-- JavaScript -->
<script>
    // Function to load text content using vanilla JavaScript
    document.getElementById('loadTextWithoutJQuery').addEventListener('click',
function() {
    var xhr = new XMLHttpRequest();
    xhr.open('GET', 'sample.txt', true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState === 4 && xhr.status === 200) {
            document.getElementById('textOutput').textContent = xhr.responseText;
        }
    };
    xhr.send();
});

// Function to load text content using jQuery
$('#loadTextWithJQuery').click(function() {
    $.ajax({
        url: 'sample.txt',
        method: 'GET',
        success: function(data) {
            $('#textOutput').text(data);
        },
        error: function() {

```

```

        $('#textOutput').text('Error loading text file.');
    }
});

// Function to load JSON data using jQuery's getJSON method
$('#loadJson').click(function() {
    $.getJSON('data.json', function(data) {
        $('#jsonOutput').text(JSON.stringify(data, null, 2));
    }).fail(function() {
        $('#jsonOutput').text('Error loading JSON file.');
    });
});

// Function to parse JSON data using jQuery's parseJSON method
$('#parseJson').click(function() {
    var jsonString = '{"name": "John", "age": 30, "city": "New York"}';
    var jsonObject = $.parseJSON(jsonString);
    $('#jsonOutput').text('Name: ' + jsonObject.name + '\nAge: ' + jsonObject.age +
    '\nCity: ' + jsonObject.city);
});
</script>
</body>
</html>

```

**Output**

The screenshot shows a web browser window with the URL 127.0.0.1:5500/program\_10.html. The page title is "AJAX Examples". Below the title are four blue buttons: "Load Text (No jQuery)", "Load Text (With jQuery)", "Load JSON", and "Parse JSON". Underneath the buttons, the heading "Text File Content" is displayed. A light gray box contains the text "Reading the contents via AJAX call without using JQuery". Below this, the heading "JSON Data" is shown above a horizontal progress bar.

AJAX Examples

Load Text (No jQuery) Load Text (With jQuery) Load JSON Parse JSON

**Text File Content**

Reading the contents via AJAX call using JQuery

**JSON Data**

```
{ "name": "John", "age": 30, "city": "New York" }
```

AJAX Examples

Load Text (No jQuery) Load Text (With jQuery) Load JSON Parse JSON

**Text File Content**

**JSON Data**

```
{ "name": "John", "age": 30, "city": "New York" }
```

AJAX Examples

Load Text (No jQuery) Load Text (With jQuery) Load JSON Parse JSON

**Text File Content**

**JSON Data**

```
Name: John  
Age: 30  
City: New York
```

**Explanation:****1. HTML:**

- The HTML structure includes buttons for different AJAX operations and output areas to display results.
- There are buttons for loading text and JSON data using both vanilla JavaScript and jQuery, as well as for parsing JSON.

**2. CSS:**

- Basic styling is applied to the body, buttons, and output areas to make the page visually appealing.

**3. JavaScript:****○ Without jQuery:**

- The `loadTextWithoutJQuery` button uses vanilla JavaScript's `XMLHttpRequest` to load and display text from a file (`sample.txt`).

**○ With jQuery:**

- The `loadTextWithJQuery` button uses jQuery's `$.ajax()` method to load and display text from a file.
- The `loadJson` button uses jQuery's `$.getJSON()` method to fetch and display JSON data from `data.json`.
- The `parseJson` button demonstrates `$.parseJSON()` by parsing a JSON string and displaying its contents.

**Files Needed:**

1. `sample.txt`: A text file containing sample text. For example:

**Content of sample.txt file**

This is some sample text loaded from a text file.

2. `data.json`: A JSON file containing sample JSON data. For example:

**Content of data.json file**

```
{  
    "name": "John",  
    "age": 30,  
    "city": "New York"  
}
```

You can save the provided HTML code into an `index.html` file and ensure that `sample.txt` and `data.json` are in the same directory as your HTML file. Open `index.html` in a web browser to see the examples in action.

Remember that AJAX requests need to be served from a web server or local server environment to work correctly (i.e., they might not work if just opened directly from the file system due to browser security restrictions). You can use tools like Live Server in VS Code or a simple local server like `http-server` in Node.js for testing.

## **VIVA QUESTIONS**

### **HTML**

**1. What is HTML, and what does it stand for?**

- HTML stands for HyperText Markup Language. It is used to structure content on the web.

**2. What is the basic structure of an HTML document?**

- The basic structure includes <!DOCTYPE html>, <html>, <head>, and <body> tags.

**3. What is the purpose of the <head> section in an HTML document?**

- It contains meta-information, links to stylesheets, scripts, and the title of the document.

**4. What is the purpose of the <title> tag?**

- The <title> tag sets the title of the webpage, which appears in the browser tab.

**5. What is an HTML element?**

- An HTML element is a component of an HTML document, consisting of an opening tag, content, and a closing tag.

**6. What is an attribute in HTML?**

- An attribute provides additional information about an HTML element and is included in the opening tag. It usually consists of a name and a value (e.g., src="image.jpg").

**7. What is the difference between block-level and inline elements?**

- Block-level elements (e.g., <div>, <h1>) take up the full width available and start on a new line. Inline elements (e.g., <span>, <a>) only take up as much width as necessary and do not start on a new line.

**8. How do you create a hyperlink in HTML?**

- Use the <a> tag with the href attribute to specify the URL. Example: <a href="https://example.com">Visit Example</a>.

**9. What does the target="\_blank" attribute do in a hyperlink?**

- It opens the linked document in a new tab or window.

**10. How do you include an image in an HTML document?**

- Use the <img> tag with the src attribute to specify the image source and the alt attribute for alternative text. Example: .

**11. What is the purpose of the alt attribute in the <img> tag?**

- The alt attribute provides alternative text for the image if it cannot be displayed. It also improves accessibility.

**12. What is an unordered list, and how is it created?**

- An unordered list is a list where items are not numbered. It is created using the `<ul>` tag, with each item marked by `<li>` tags. Example:  
`<ul>`  
    `<li>Item 1</li>`  
    `<li>Item 2</li>`  
`</ul>`

**13. How do you create an ordered list in HTML?**

- Use the `<ol>` tag to create an ordered list, with each item marked by `<li>` tags.  
Example:  
`<ol>`  
    `<li>First item</li>`  
    `<li>Second item</li>`  
`</ol>`

**14. What is the purpose of the `<form>` tag in HTML?**

- The `<form>` tag is used to create a form for collecting user input.

**15. How do you create a text input field in a form?**

- Use the `<input>` tag with the `type="text"` attribute. Example: `<input type="text" name="username">`.

**16. What is the `<textarea>` element used for?**

- The `<textarea>` element is used to create a multi-line text input field.

**17. How do you create a dropdown menu in HTML?**

- Use the `<select>` tag with `<option>` tags for each dropdown item. Example:  
`<select name="options">`  
    `<option value="1">Option 1</option>`  
    `<option value="2">Option 2</option>`  
`</select>`

**18. What is the `<button>` element used for?**

- The `<button>` element is used to create a clickable button. It can be used within forms or as standalone buttons.

**19. How do you create a table in HTML?**

- Use the `<table>` tag with `<tr>` for rows, `<th>` for header cells, and `<td>` for data cells. Example:  
`<table>`  
    `<tr>`  
        `<th>Header 1</th>`  
        `<th>Header 2</th>`  
    `</tr>`  
    `<tr>`  
        `<td>Data 1</td>`  
        `<td>Data 2</td>`  
    `</tr>`  
`</table>`

**20. What is the <caption> tag used for in a table?**

- The <caption> tag provides a title or description for the table.

**21. What is the purpose of the <meta> tag?**

- The <meta> tag provides metadata about the HTML document, such as character encoding and viewport settings.

**22. How do you specify the character encoding of an HTML document?**

- Use the <meta charset="UTF-8"> tag in the <head> section.

**23. What is the <link> tag used for?**

- The <link> tag is used to link external resources, such as stylesheets, to the HTML document.

**24. How do you include an external CSS file in an HTML document?**

- Use the <link> tag with rel="stylesheet" and href attributes. Example: <link rel="stylesheet" href="styles.css">.

**25. What is the <script> tag used for in HTML?**

- The <script> tag is used to include JavaScript code or link to external JavaScript files.

**26. How do you add comments in HTML?**

- Use the <!-- comment --> syntax to add comments.

**27. What is the <div> element used for?**

- The <div> element is used as a container to group and style sections of a webpage.

**28. What is the <span> element used for?**

- The <span> element is used to group and style inline portions of text or other elements.

**29. What are semantic HTML elements?**

- Semantic HTML elements clearly describe their content and purpose, such as <header>, <footer>, <article>, and <section>.

**30. How do you create a line break in HTML?**

- Use the <br> tag to insert a line break.

**31. What is the difference between the <strong> and <b> tags?**

- The <strong> tag indicates that text is of strong importance, while <b> simply applies bold styling without implying importance.

**32. What does the class attribute do in HTML?**

- The class attribute assigns one or more class names to an element, which can be used for styling with CSS or targeting with JavaScript.

**33. What does the id attribute do in HTML?**

- The id attribute assigns a unique identifier to an element, which can be used for styling, JavaScript targeting, or linking.

**34. How do you create a checkbox in HTML?**

- Use the <input> tag with the type="checkbox" attribute. Example: <input type="checkbox" name="accept" value="yes">.

**35. How do you create a radio button in HTML?**

- Use the <input> tag with the type="radio" attribute. Example: <input type="radio" name="gender" value="male"> Male.

**36. What is the <iframe> tag used for?**

- The <iframe> tag is used to embed another HTML document within the current document.

**37. How do you make a form submit to a specific URL?**

- Use the action attribute of the <form> tag to specify the URL where the form data will be sent. Example: <form action="submit.php">.

**38. What does the method attribute do in a form?**

- The method attribute specifies how the form data should be sent to the server. Common values are GET and POST.

**39. How do you make text italic in HTML?**

- Use the <i> or <em> tags to make text italic. <em> is semantically preferred as it indicates emphasis.

**40. What is the data-\* attribute used for in HTML?**

- The data-\* attribute is used to store custom data that can be accessed using JavaScript. It allows embedding of custom information into HTML elements.

---

## Stylesheets – CSS

**1. What is CSS, and what does it stand for?**

- CSS stands for Cascading Style Sheets. It is used to control the presentation, layout, and design of HTML elements on a webpage.

**2. How can CSS be included in an HTML document?**

- CSS can be included in three ways:
  - **Inline CSS:** Using the style attribute within HTML elements (e.g., <p style="color: red;">).
  - **Internal CSS:** Using a <style> tag within the <head> section of the HTML document.
  - **External CSS:** Linking to a separate CSS file using a <link> tag in the <head> section.

**3. What is the syntax for a CSS rule?**

- A CSS rule consists of a selector and a declaration block. Example:

```
selector {  
    property: value;  
}
```

For example:

```
p {  
    color: blue;  
}
```

**4. What is the purpose of the class selector in CSS?**

- The class selector is used to apply styles to elements with a specific class attribute. It is defined with a dot before the class name (e.g., .my-class).

**5. How does the id selector differ from the class selector in CSS?**

- The id selector targets a single unique element with a specific id attribute, while the class selector can target multiple elements with the same class. The id selector is defined with a hash symbol (e.g., #my-id), and the class selector is defined with a dot (e.g., .my-class).

**6. What is specificity in CSS?**

- Specificity determines which CSS rule is applied when multiple rules match the same element. It is calculated based on the types of selectors used (e.g., inline styles, IDs, classes, and elements).

**7. What is the !important declaration in CSS?**

- The !important declaration is used to give a CSS rule higher priority and override other rules. It should be used sparingly. Example: color: red !important;

**8. What are pseudo-classes in CSS?**

- Pseudo-classes are keywords added to selectors that specify a special state of the selected elements. Examples include :hover, :focus, and :nth-child.

**9. What are pseudo-elements in CSS?**

- Pseudo-elements are used to style specific parts of an element. Examples include ::before, ::after, and ::first-line.

**10. What is the box model in CSS?**

- The box model describes the rectangular boxes generated for elements in the document tree. It includes content, padding, border, and margin areas.

**11. How do you center a block-level element horizontally using CSS?**

- Set the margin property to auto and specify a width for the element. Example:

```
.centered {  
    width: 50%;  
    margin: 0 auto;  
}
```

**12. What is the difference between padding and margin in CSS?**

- Padding is the space between the content of an element and its border. Margin is the space outside the border, separating the element from other elements.

**13. How do you apply a style to multiple elements using CSS?**

- List the elements separated by commas in the selector. Example:

```
h1, h2, h3 {  
    color: green;  
}
```

**14. What is the display property in CSS, and what are its common values?**

- The display property specifies how an element is displayed on the page. Common values include block, inline, inline-block, none, and flex.

**15. How do you create a responsive design using CSS?**

- Use media queries to apply different styles for different screen sizes and devices. Example:

```
@media (max-width: 600px) {  
    body {  
        background-color: lightblue;  
    }  
}
```

**16. What is Flexbox in CSS?**

- Flexbox is a layout model that allows you to design complex layouts with flexible and responsive arrangements. It uses properties like display: flex, justify-content, align-items, and flex-direction.

**17. What is CSS Grid Layout?**

- CSS Grid Layout is a two-dimensional layout system for the web that allows you to create complex grid-based designs with rows and columns. It uses properties like display: grid, grid-template-rows, and grid-template-columns.

**18. What does the float property do in CSS?**

- The float property is used to position an element to the left or right within its containing block, allowing other content to wrap around it.

**19. What is the position property in CSS, and what are its values?**

- The position property specifies how an element is positioned in the document. Common values include static, relative, absolute, fixed, and sticky.

**20. How do you use the z-index property in CSS?**

- The z-index property specifies the stack order of elements. Elements with a higher z-index value are displayed in front of elements with a lower value. It only works on positioned elements (position set to relative, absolute, fixed, or sticky).

**21. What is the opacity property in CSS?**

- The opacity property sets the transparency level of an element, with values ranging from 0 (completely transparent) to 1 (completely opaque).

**22. How do you change the font size of an element using CSS?**

- Use the font-size property. Example: font-size: 16px;.

**23. What is the background shorthand property in CSS?**

- The background shorthand property allows you to set multiple background properties in one declaration, such as background-color, background-image, background-repeat, and background-position.

**24. What is the purpose of the border-radius property in CSS?**

- The border-radius property is used to create rounded corners on elements.

**25. How do you apply a gradient background using CSS?**

- Use the background-image property with a linear-gradient or radial-gradient function. Example:  
background-image: linear-gradient(to right, red, yellow);

**26. What are media queries in CSS?**

- Media queries are used to apply different styles based on the characteristics of the device, such as screen width, height, or orientation.

**27. How do you set the width of an element to 100% of its parent container?**

- Use width: 100%; in the CSS rule for the element.

**28. What is the text-align property used for?**

- The text-align property is used to set the horizontal alignment of text within an element. Common values include left, right, center, and justify.

**29. What is the line-height property in CSS?**

- The line-height property specifies the amount of space between lines of text.

**30. How do you remove the default underline from links using CSS?**

- Use the text-decoration property with the value none. Example: a { text-decoration: none; }.

**31. What is the box-shadow property used for?**

- The box-shadow property adds shadow effects around an element's frame.  
Example:  
box-shadow: 2px 2px 5px grey;

**32. How do you change the font family of an element using CSS?**

- Use the font-family property. Example: font-family: Arial, sans-serif;.

**33. What is the text-transform property in CSS?**

- The text-transform property is used to control the capitalization of text. Values include uppercase, lowercase, and capitalize.

**34. How do you use the :hover pseudo-class in CSS?**

- The :hover pseudo-class applies styles when the user hovers over an element with their mouse. Example:

```
a:hover {
  color: red;
}
```

**35. What is the cursor property in CSS?**

- The cursor property specifies the type of cursor to be displayed when hovering over an element. Common values include pointer, default, and wait.

**36. How do you apply different styles to even and odd table rows?**

- Use the :nth-child pseudo-class with the even or odd keyword. Example:

```
tr:nth-child(even) {
  background-color: #f2f2f2;
}
```

**37. What is the visibility property in CSS, and how does it differ from display?**

- The visibility property controls whether an element is visible or not, with values visible and hidden. Unlike display: none, visibility: hidden keeps the element in the layout but hides it

## PHP

**1. What is PHP?**

- PHP (Hypertext Preprocessor) is a widely-used, open-source server-side scripting language designed primarily for web development. It can be embedded into HTML and interacts with databases to generate dynamic content.

**2. How does PHP handle form data?**

- PHP handles form data through the \$\_GET and \$\_POST superglobals. \$\_GET is used for retrieving data sent via HTTP GET method, while \$\_POST is used for data sent via HTTP POST method.

**3. What are the different types of variables in PHP?**

- PHP supports several types of variables including:
  - **Integers:** Whole numbers.
  - **FLOATS:** Decimal numbers.
  - **Strings:** Textual data.
  - **Booleans:** true or false.
  - **Arrays:** Collections of values.
  - **Objects:** Instances of classes.
  - **NULL:** Represents a variable with no value.

**4. What is the difference between == and === in PHP?**

- == is the equality operator that checks if two values are equal after type juggling (type conversion). === is the identity operator that checks if two values are equal and of the same type.

**5. How do you define a function in PHP?**

- A function in PHP is defined using the function keyword.
- ```
function myFunction($param) {
    // function body
}
```

**6. What is the purpose of the include and require statements in PHP?**

- Both include and require are used to include the content of one PHP file into another. The main difference is that require will produce a fatal error and stop script execution if the file is not found, whereas include will only produce a warning and continue execution.

**7. What are PHP superglobals?**

- Superglobals are built-in global arrays in PHP that are always accessible, regardless of scope. Examples include:
  - \$\_GET, \$\_POST, \$\_REQUEST, \$\_SESSION, \$\_COOKIE, \$\_FILES, \$\_SERVER, \$\_ENV, \$\_GLOBALS.

**8. How do you connect to a MySQL database in PHP?**

- You can use the mysqli or PDO extension to connect to a MySQL database.

Example using mysqli:

```
$conn = new mysqli("hostname", "username", "password", "database");
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

**9. How do you execute a SQL query in PHP?**

- To execute a SQL query, you use the query method of the mysqli object or exec method of the PDO object.

```
$sql = "SELECT * FROM table";
$result = $conn->query($sql);
```

```
if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        // Process each row
    }
}
```

**10. How do you handle errors in PHP?**

- PHP provides various methods for error handling, including:
  - **Error reporting functions:** error\_reporting(), set\_error\_handler().
  - **Exception handling:** Using try, catch, and finally blocks to handle exceptions.

```
try {
    // Code that may throw an exception
} catch (Exception $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
}
```

**11. What is SQL injection and how can you prevent it?**

- SQL injection is a security vulnerability where an attacker can manipulate SQL queries by injecting malicious input. To prevent it, use prepared statements and parameterized queries:  

```
$stmt = $conn->prepare("SELECT * FROM users WHERE username = ?");  
$stmt->bind_param("s", $username);  
$stmt->execute();
```

**12. How do you manage sessions in PHP?**

- Sessions are managed using the `$_SESSION` superglobal. Start a session with `session_start()` and use `$_SESSION` to store session data.  

```
session_start();  
$_SESSION['username'] = 'JohnDoe';
```

**13. How do you upload a file in PHP?**

- File uploads are handled through the `$_FILES` superglobal. You use the `move_uploaded_file()` function to move the uploaded file from the temporary directory to a desired location.  

```
if (isset($_FILES['file'])) {  
    $file = $_FILES['file'];  
    $destination = 'uploads/' . $file['name'];  
    move_uploaded_file($file['tmp_name'], $destination);  
}
```

**14. How do you read and write files in PHP?**

- Use functions like `fopen()`, `fread()`, `fwrite()`, and `fclose()` to handle file operations.  

```
$file = fopen("example.txt", "r");  
$content = fread($file, filesize("example.txt"));  
fclose($file);  
  
$file = fopen("example.txt", "w");  
fwrite($file, "New content");  
fclose($file);
```

## JavaScript

### **1. What is JavaScript and what is it used for?**

- JavaScript is a high-level, interpreted programming language primarily used to create dynamic and interactive content on websites. It allows for client-side scripting to enhance user experiences.

### **2. Explain the difference between `var`, `let`, and `const`.**

- `var` is function-scoped and can be redeclared. `let` is block-scoped and cannot be redeclared within the same block. `const` is also block-scoped but is used for variables that should not be reassigned after initialization.

### **3. What are data types in JavaScript?**

- JavaScript has primitive data types like `number`, `string`, `boolean`, `undefined`, `null`, `symbol`, and `bignum`. There are also non-primitive types like `object` and `array`.

### **4. What is the difference between `==` and `===`?**

- `==` (loose equality) compares values after performing type coercion if necessary, while `===` (strict equality) compares both value and type without type coercion.

### **5. How does prototypal inheritance work in JavaScript?**

- JavaScript uses prototypes for inheritance. Each object has a prototype object from which it can inherit properties and methods. This is achieved through the prototype chain.

### **6. What is the `this` keyword in JavaScript?**

- `this` refers to the context in which a function is executed. It can vary depending on how the function is called: it refers to the global object in non-strict mode, the object that is calling the function in method calls, or it can be explicitly bound using `call`, `apply`, or `bind`.

### **7. What are JavaScript prototypes and how do you use them?**

- Prototypes are a mechanism by which JavaScript objects inherit features from one another. You can set the prototype of an object using `Object.create()` or by setting the `prototype` property of a constructor function.

### **8. What are callbacks in JavaScript?**

- Callbacks are functions passed as arguments to other functions and are executed after the completion of the other function. They are used to handle asynchronous operations.

### **9. Explain promises and how they are used.**

- Promises represent the eventual completion (or failure) of an asynchronous operation and its resulting value. They provide a cleaner way to handle asynchronous code compared to callbacks, using methods like `then()`, `catch()`, and `finally()`.

**10. What is async/await and how does it improve asynchronous code?**

- async and await are syntactic sugar built on top of promises, making asynchronous code look and behave more like synchronous code. async functions return a promise, and await pauses the function execution until the promise is resolved.

**11. What are JavaScript modules and why are they used?**

- JavaScript modules are a way to organize code by exporting and importing functions, objects, or primitives from one file to another. This improves code maintainability and encapsulation. Modules are typically managed using the import and export keywords.

**12. Explain event delegation in JavaScript.**

- Event delegation is a technique where a single event listener is added to a parent element to manage events for its child elements. This leverages event bubbling and can be more efficient than adding multiple listeners to individual elements.

**13. What are the different ways to handle errors in JavaScript?**

- Errors can be handled using try, catch, finally, and throw. try contains the code that might throw an error, catch handles the error, and finally executes code regardless of the result. throw is used to manually trigger an error.

## AJAX

**1. What is AJAX and what are its primary components?**

- AJAX stands for Asynchronous JavaScript and XML. It allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. Its primary components include JavaScript, the XMLHttpRequest object, and often XML or JSON for data interchange.

**2. How does AJAX improve user experience on web pages?**

- AJAX improves user experience by enabling web pages to update content dynamically without needing a full page reload. This results in a smoother, faster, and more responsive user experience.

**3. Can you explain the process of making an AJAX request?**

- Making an AJAX request typically involves creating an instance of the XMLHttpRequest object, setting up the request (specifying the HTTP method, URL, and whether the request should be asynchronous), sending the request, and handling the server's response in a callback function.

**4. What are the key methods and properties of the XMLHttpRequest object?**

- Key methods include `open()`, `send()`, `setRequestHeader()`, and `abort()`. Key properties include `readyState`, `status`, and `responseText` or `responseXML`.

**5. What is the purpose of the readyState property in XMLHttpRequest?**

- The readyState property indicates the current state of the XMLHttpRequest object. It can have values from 0 (uninitialized) to 4 (complete), which correspond to different stages of the request lifecycle.

**6. How can you handle errors in an AJAX request?**

- Errors can be handled by checking the status property of the XMLHttpRequest object in the onreadystatechange callback function. You can also use the onerror event handler to catch network-level errors.

**7. What is JSON, and how is it used with AJAX?**

- JSON (JavaScript Object Notation) is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. In AJAX, JSON is often used as the format for sending and receiving data between the client and server due to its simplicity and ease of use with JavaScript.

**8. Explain the difference between synchronous and asynchronous requests in AJAX.**

- In synchronous requests, the browser waits for the server response before continuing with the execution of the script, which can lead to a frozen user interface. In asynchronous requests, the browser can continue to process other tasks while waiting for the server response, providing a more fluid user experience.

**9. What are some common uses of AJAX in web development?**

- Common uses include form submission without reloading the page, loading new content dynamically (e.g., infinite scroll), updating specific parts of a page (e.g., live search suggestions), and interacting with APIs to fetch or send data.

**10. How does AJAX differ from traditional full-page form submission?**

- Traditional form submission involves sending the entire form data to the server and refreshing the page with the server's response. AJAX allows for sending and receiving data asynchronously without reloading the page, which enables partial updates and a more seamless user experience.

## **JQUERY**

**1. What is jQuery, and why is it used in web development?**

- jQuery is a fast, lightweight, and feature-rich JavaScript library designed to simplify HTML document traversal, event handling, animation, and Ajax interactions. It is used to streamline and simplify tasks that would otherwise require more verbose JavaScript code.

**2. How do you include jQuery in a web page?**

- jQuery can be included in a web page by linking to a jQuery CDN (Content Delivery Network) or by downloading and hosting the jQuery file locally. The

typical inclusion in HTML is done with a `<script>` tag, either referencing the jQuery library from a CDN or from a local file:

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js">
</script>
or
<script src="path/to/your/local/jquery.min.js"></script>
```

### 3. What is the purpose of the `$` function in jQuery?

- The `$` function is a shorthand for the `jQuery` function. It is used to select elements, create new `jQuery` objects, and call `jQuery` methods.
- For example, `$( '#myElement' )` selects an element with the ID `myElement`.

### 4. How do you handle events in jQuery?

- Events in `jQuery` are handled using the `.on()` method or its shorthand `.click()`, `.hover()`, etc.

For example:

```
$('#myButton').on('click', function() {
    alert('Button clicked!');
});
or
$('#myButton').click(function() {
    alert('Button clicked!');
});
```

### 5. What is the difference between `.hide()` and `.fadeOut()` in jQuery?

- `.hide()` immediately hides the selected elements by setting their `display` property to `none`. `.fadeOut()` gradually hides the elements by animating the opacity from visible to hidden, creating a fading effect.

### 6. How can you make an AJAX request using jQuery?

- `jQuery` provides several methods for making AJAX requests, such as `.ajax()`, `.get()`, and `.post()`.

For example:

```
$.ajax({
    url: 'example.com/data',
    method: 'GET',
    success: function(response) {
        console.log(response);
    },
    error: function(error) {
        console.log('Error:', error);
    }
});
```

### 7. What is chaining in jQuery, and how does it work?

- Chaining in `jQuery` allows you to perform multiple operations on the same `jQuery` object in a single line of code. It works by returning the `jQuery` object itself from a method, enabling subsequent methods to be called. For example:

```
$('#myElement').css('color',
'red').slideUp(2000).slideDown(2000);
```

### 8. How do you traverse the DOM with jQuery?

- `jQuery` provides various methods for traversing the DOM, such as `.parent()`, `.children()`, `.find()`, `.siblings()`, and `.prev()`.

For example:

```
$('#myElement').parent(); // Selects the parent element  
$('#myElement').find('.child'); // Selects child elements with  
class 'child'
```

#### 9. What is the purpose of the .data() method in jQuery?

- The .data() method is used to store and retrieve data associated with DOM elements. It provides a way to attach arbitrary data to elements that can be accessed later.

For example:

```
$('#myElement').data('key', 'value'); // Store data  
var value = $('#myElement').data('key'); // Retrieve data
```

#### 10. How do you handle form submission with jQuery?

- You can handle form submission using the .submit() event handler. You can prevent the default form submission behavior and perform actions like AJAX requests:

```
$('#myForm').on('submit', function(event) {  
    event.preventDefault(); // Prevent default form submission  
    // Perform AJAX request or other actions  
});
```